AMENDMENTS TO THE CLAIMS:

This listing of claims replaces all prior versions and listings of claims in the application:

LISTING OF CLAIMS:

Claims 1 - 17. Cancelled

18. (Currently Amended) A method of controlling access to a Static Random Access Memory (SRAM), the method comprising:

~~providing~~ storing, in a set of queues, ~~that store~~ queue entries for requests received from multiple programmable processors, the set including:

a ~~read~~ first queue to queue read requests; and

~~an order~~ a second queue to ~~queue ordered requests~~ store memory reference requests, the second queue maintaining the order of the memory reference requests; and

consecutively servicing chained requests in a one of the queues before servicing another one of the queues.

19. (Currently Amended) The method of claim 18, wherein the set of queues comprises a read lock fail queue to ~~store queue entries for requests to locked memory~~ hold read memory reference requests that fail because of a lock existing on a portion of memory.

20. (Previously Presented) The method of claim 19, further comprising consecutively servicing requests in the read lock fail queue until an entry is encountered that remains locked.

21. (Previously Presented) The method of claim 18, wherein the set of queues further comprises a read/write queue to store queue entries for memory requests received from a core processor.

22. (Previously Presented) The method of claim 18, further comprising receiving packets via at least one media access controller (MAC).

23. (New) The method of claim 18, wherein the first queue is a read queue and the second queue is an order queue.

24. (New) The method of claim 18, further comprising storing contiguous memory references in the second queue, the contiguous memory references including multiple memory references from a single thread.

25. (New) The method of claim 18, wherein the memory reference requests stored in the second queue comprise writes to SRAM and reads that are to be non-optimized.

26. (New) A system for controlling access to a Static Random Access Memory (SRAM), the system comprising:

a set of queues that store queue entries for requests received from multiple programmable processors, the set including:

a first queue to queue read requests; and

a second queue to store memory reference requests, the second queue maintaining the order of the memory reference requests; and

one or more processors configured to consecutively service chained requests in one of the queues before servicing another one of the queues.

27. (New) The system of claim 26, wherein the first queue is a read queue and the second queue is an order queue.

28. (New) The system of claim 26, wherein the set of queues comprises a read lock fail queue to hold read memory reference requests that fail because of a lock existing on a portion of memory.

29. (New) The system of claim 28, wherein the one or more processors are further configured to consecutively service requests in the read lock fail queue until an entry is encountered that remains locked.

30. (New) The system of claim 26, wherein the set of queues further comprises a read/write queue to store queue entries for memory requests received from a core processor.

31. (New) The system of claim 26, wherein the second queue is configured to store contiguous memory references, the contiguous memory references comprising multiple memory references from a single thread.

32. (New) The system of claim 26, wherein the memory reference requests stored in the second queue comprise writes to SRAM and reads that are to be non-optimized.

33. (New) An article for controlling access to a Static Random Access Memory (SRAM), the article comprising one or more machine-readable media to store instructions that are executable to cause one or more processing devices to:

store, in a set of queues, queue entries for requests received from multiple programmable processors, the set including:

a first queue to queue read requests; and

a second queue to store memory reference requests, the second queue maintaining the order of the memory reference requests; and

consecutively service chained requests in a one of the queues before servicing another one of the queues.

34. (New) The article of claim 33, wherein the first queue is a read queue and the second queue is an order queue.

35. (New) The article of claim 33, wherein the set of queues comprises a read lock fail queue, the computer program product further comprising instructions to:

hold, in the read lock fail queue, read memory reference requests that fail because of a lock existing on a portion of memory.

36. (New) The article of claim 35, further comprising instructions to:

consecutively service requests in the read lock fail queue until an entry is encountered that remains locked.

37. (New) The article of claim 33, wherein the set of queues further comprises a read/write queue, the computer program product further comprising instructions to:

store, in the read/write queue, queue entries for memory requests received from a core processor.

38. (New) The article of claim 33, further comprising instructions to:

receive packets via at least one media access controller (MAC).

39. (New) The article of claim 33, further comprising instructions to:

store contiguous memory references in the second queue, the contiguous memory references including multiple memory references from a single thread.

40. (New) The article of claim 33, wherein the memory reference requests stored in the second queue comprise writes to SRAM and reads that are to be non-optimized.